Trajectory-based machine learning method and its application to molecular dynamics

Han, Ruocheng ; Luber, Sandra

Abstract: Ab initio molecular dynamics (AIMD) has become a popular simulation technique but long simulation times are often hampered due to its high computational effort. Alternatively, classical molecular dynamics (MD) based on force fields may be used, which, however, has certain shortcomings compared to AIMD. In order to alleviate that situation, a trajectory-based machine learning (TrajML) approach is introduced for the construction of force fields by learning from AIMD trajectories. Only nuclear trajectories are required, which can be obtained by other methods beyond AIMD as well. We developed an easy-to-use MD machine learning package (TrajML MD) for instant modelling of the force field and system-focussed prediction of molecular configurations for MD trajectories. It consumes similar computational resources as classical MD but can simulate complex systems with a higher accuracy due to the targeted learning on the system of interest.

DOI: https://doi.org/10.1080/00268976.2020.1788189

RESEARCH ARTICLE

Trajectory-Based Machine Learning method and its application to Molecular Dynamics

R. Han[a] and S. Luber[a]

[a]Department of Chemistry A, University of Zurich, Winterthurerstrasse 190, 8057 Zurich, Switzerland

ARTICLE HISTORY
Compiled June 7, 2020

ABSTRACT
Ab initio molecular dynamics (AIMD) has become a popular simulation technique but long simulation times are often hampered due to its high computational effort. Alternatively, classical molecular dynamics (MD) based on force fields may be used, which, however, has certain shortcomings compared to AIMD. In order to alleviate that situation, a trajectory-based machine learning (TrajML) approach is introduced for the construction of force fields by learning from AIMD trajectories. Only nuclear trajectories are required, which can be obtained by other methods beyond AIMD as well. We developed an easy-to-use MD machine learning package (TrajML MD) for instant modeling of the force field and system-focused prediction of molecular configurations for MD trajectories. It consumes similar computational resources as classical MD but can simulate complex systems with a higher accuracy due to the targeted learning on the system of interest.

KEYWORDS
Machine learning; force fields; molecular dynamics; *ab initio* molecular dynamics; instant modeling

1. Introduction

Molecular dynamics (MD) as an important simulation technique can provide an understanding of dynamic properties of the system under study at an atomistic level. A crucial aspect for MD is the simulation length. Prediction of thermodynamic properties, analysis of chemical processes and computation of spectroscopic signatures, for instance, require certain time scales of the MD simulations. Due to the long time-scales required, MD simulations are usually based on classical force fields generated by fitting *ab initio* results to some human-designed empirical terms, e.g. bond stretching, angle bending, dihedral/improper dihedral torsion, Van der Waals interaction, and electrostatic interaction. Although they are widely applied, many cases of failures have been published, such as non-tolerable errors in dihedral rotation [1] and hydrogen bonding [2]. Moreover, weak interactions and charged or transition-metal based systems are difficult to model. While density functional theory (DFT)-based [3, 4] MD (usually called *ab initio* MD (AIMD)) in general provides more accurate trajectories than classical MD, the computational cost at a reasonable time scale is often high and may thus not be possible to

CONTACT S. Luber Email: sandra.luber@chem.uzh.ch

accomplish.

During recent years, machine learning has become more and more popular. It has been applied to various areas such as computer vision, natural language processing, self-driving and artificial intelligence. In computational chemistry, machine learning force fields or potentials (MLFFs/MLPs) [5–40] are emerging as a powerful approach. These MLFFs/MLPs use artificial neural networks (ANNs), Gaussian process regressions, kernel-based regression, or other approaches to fit their features to energies and/or forces of DFT or wavefunction-based calculations. Among them, several MLFFs/MLPs have been especially designed for molecular systems. ANI [19] and TensorMol [23] use Behler–Parrinello type neural networks [5] which are constructed in a feed-forward way. DTNN [15], HIP-NN [24], SchNet [25], and PhysNet [26] apply a message-passing scheme [41] in their neural networks so that the information propagation is guided by the graph structure. Atom types, positions, charges, and molecular topologies are provided as input and large databases are used for training in the above mentioned ANN-based approaches. Besides, FFLUX [16–18] constructs atomic multipole moments as descriptors in a Gaussian process regression to provide a full description of the electronic information in molecules.

The main focus of these mentioned methods is on the transferability of models by constructing MLFFs/MLPs based on large databases containing molecules and their conformers. Nevertheless, in the case of MD simulations, they still have limitations on the systems modelled. Complex systems containing e.g. non-zero charges or unusual atom types (such as metal atoms and their various spin states) are normally not simulated due to the limitation of the underlying training datasets.

To account for the cases when the systems of interest are not fully parameterized by any type of force field, we introduce a different type of MLFF – TrajML FF – and apply it to MD simulations. TrajML FF does not depend on any existing database but only on the MD trajectory(ies) of the system itself, providing a system-focused instant parameterization. In this work, we generated the required trajectories with AIMD. This can be adopted further for longer simulations which would e.g. not be feasible otherwise due to the high computational effort. Reduction of computational effort has also been aimed at in previous works [12, 14, 20–22, 30, 34, 38] using machine learning. Some studies [14, 34, 38] have been focused on condensed phase systems and properties with on-the-fly modification of the force fields. Other studies [12, 30] have been designed mainly for intermolecular forces in solvated systems. Similar to GDML/sGDML [20–22], the TrajML FF we developed aims at intramolecular forces within molecular systems. While the ANN-based approaches have in general the problem of explainability, TrajML is encoded explicitly with the power series of distances, making it a concise force field with more physical meaning. During finishing this manuscript, we became aware of other works (Refs. [42, 43]) discussing or applying distance-based power series for ML, however not with the focus on the training and prediction of MD trajectories. TrajML FF is designed for force-based input, with only the geometry information required for the training data. This makes the TrajML a convenient approach for the instant modeling of force fields and the system-focused prediction of molecular configurations for MD trajectories.

2. Theory

In the following, the underlying theory for the TrajML is shortly introduced.

2.1. Numerical integration algorithm

Numerical integration algorithms are used for the calculation of trajectories in MD simulations. In the TrajML code, the Verlet integrator [44] is used, which directly provides the relation between positions and acceleration (see Eq. 1 where $\vec{r}_i(t)$ and $\vec{a}_i(t)$ are the position and the acceleration of atom i at time t, respectively; Δt is the timestep). Besides, $\vec{a}_i(t)\Delta t^2$ in Eq. 1 is used to encode the information about the force on atom i at time t based on the fact that the term $\frac{m_i}{\Delta t^2}$ is constant for the entire simulation in our case (see Eq. 2 where $\vec{F}_i(t)$ is the force on atom i at time t, $\vec{m}_i$ is the mass of atom i, and $\vec{F}_i^{encode}(t)$ is the encoded "force" on atom i at time t).

$$\vec{r}_i(t+\Delta t) = 2\vec{r}_i(t) - \vec{r}_i(t-\Delta t) + \vec{a}_i(t)\Delta t^2 + \mathcal{O}(\Delta t^4) \tag{1}$$

$$\vec{F}_i(t) = m_i\vec{a}_i(t) = \frac{m_i}{\Delta t^2}\cdot\vec{a}_i(t)\Delta t^2 \propto \vec{a}_i(t)\Delta t^2 = \vec{F}_i^{encode}(t) \tag{2}$$

2.2. Series construction

The force between two atoms i and j ($\vec{F}_{ij}$) is described by a set of user-defined force bases, which are power series of distances between the two atoms (see Eq. 3 where f_{ij} represents the formula to be learned, s is a user-defined hyperparameter and $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$). Power series have been selected because potentials (and its derivatives) are assumed to be analytical functions and can be expanded in the form of power series. Also, some terms in the power series of the distance have a physical meaning: the force of charge–charge interactions is proportional to $\frac{1}{||\vec{r}_{ij}||^2}$, the force of charge–fixed dipole interactions is proportional to $\frac{1}{||\vec{r}_{ij}||^3}$, etc. In the prediction of the trajectories by TrajML, the power series of distances will form a Vandermonde matrix which can be iteratively generated, leading to a lower computational effort. Nevertheless, we are aware that including other many-body interactions can improve the accuracy [45, 46].

$$||\vec{F}_{ij}^{encode}|| = f_{ij}(\frac{1}{||\vec{r}_{ij}||^2}, \frac{1}{||\vec{r}_{ij}||^3}, ..., \frac{1}{||\vec{r}_{ij}||^s}) \tag{3}$$

2.3. LASSO regression

With response variable y and predictor variable $\boldsymbol{X}$ (in the form of a covariate matrix), the linear model is written as in Eq. 4, where β is the weight and ϵ is the noise. To decrease the model complexity while solving Eq. 4, the Least Absolute Shrinkage and Selection Operator [47] (LASSO) can be adopted to decrease the number of dependencies by introducing an extra penalization term in the loss function (see Eq. 5, $\hat{\beta}$ is the estimated model parameter).

The hyperparameter λ is for $L1$ regularization penalty, which can be either set manually or tuned through cross validation. The formula of minimization of the loss function is given in Eq. 6, where i is the label of data points, j is the label of covariates, and x_{ij} is an element in $\boldsymbol{X}$. Number of data points N is introduced to remove the dependence of

λ on the size of dataset. LASSO regression cannot be solved analytically and a proximal gradient descent method is used to obtain the numerical solution.

$$y = \boldsymbol{X}\beta + \epsilon \tag{4}$$

$$L_{LASSO}(\hat{\beta}) = ||y - \boldsymbol{X}\hat{\beta}||_2^2 + \lambda||\hat{\beta}||_1 \tag{5}$$

$$\begin{aligned} \hat{\beta}_{LASSO} &= \underset{\hat{\beta}}{\operatorname{argmin}}\, L_{LASSO}(\hat{\beta}) \\ &= \underset{\hat{\beta}}{\operatorname{argmin}}\{||y - \boldsymbol{X}\hat{\beta}||_2^2 + \lambda||\hat{\beta}||_1\} \\ &= \underset{\hat{\beta}}{\operatorname{argmin}}\{\frac{1}{N}\sum_i [y_i - \sum_j x_{ij}\hat{\beta}_j]^2 + \lambda\sum_j |\hat{\beta}_j|\} \end{aligned} \tag{6}$$

2.4. K-fold cross validation

Cross validation [48] (CV) is a statistical method used to evaluate machine learning models. K-fold CV splits the data set into K folds where $K-1$ folds are used for the training and one fold is used for the validation. The validation fold is iterated among K folds so that the best model for all folds can be found. In the TrajML, K-fold CV is used for the LASSO regression to automatically search for the best hyperparameter λ [49]. The hyperparameter λ is first fixed and the estimated model parameter $\hat{\beta}$ is optimized within each fold. λ is then tuned with grid search in order to produce averagely the best result for all folds.

3. Methods

An algorithm is shown in Fig. 1 for the description of the TrajML MD code. The code consists of five sections: data generation, pre-processing, machine learning, post-processing, and prediction.

3.1. Data generation

The trajectory of an AIMD simulation in the NVE ensemble is used as input data for the code. Alternatively, multiple AIMD simulations can be performed with different initial structures and combined for the input data. Strategies like metadynamics [50] and replica exchange molecular dynamics [51] can also be used for the generation of the training datasets.

3.2. Pre-processing

The pre-processing section constructs the training data and training label from the given trajectories. Firstly, the Verlet scheme is used to calculate the force $\vec{F}_i(t)$ of each

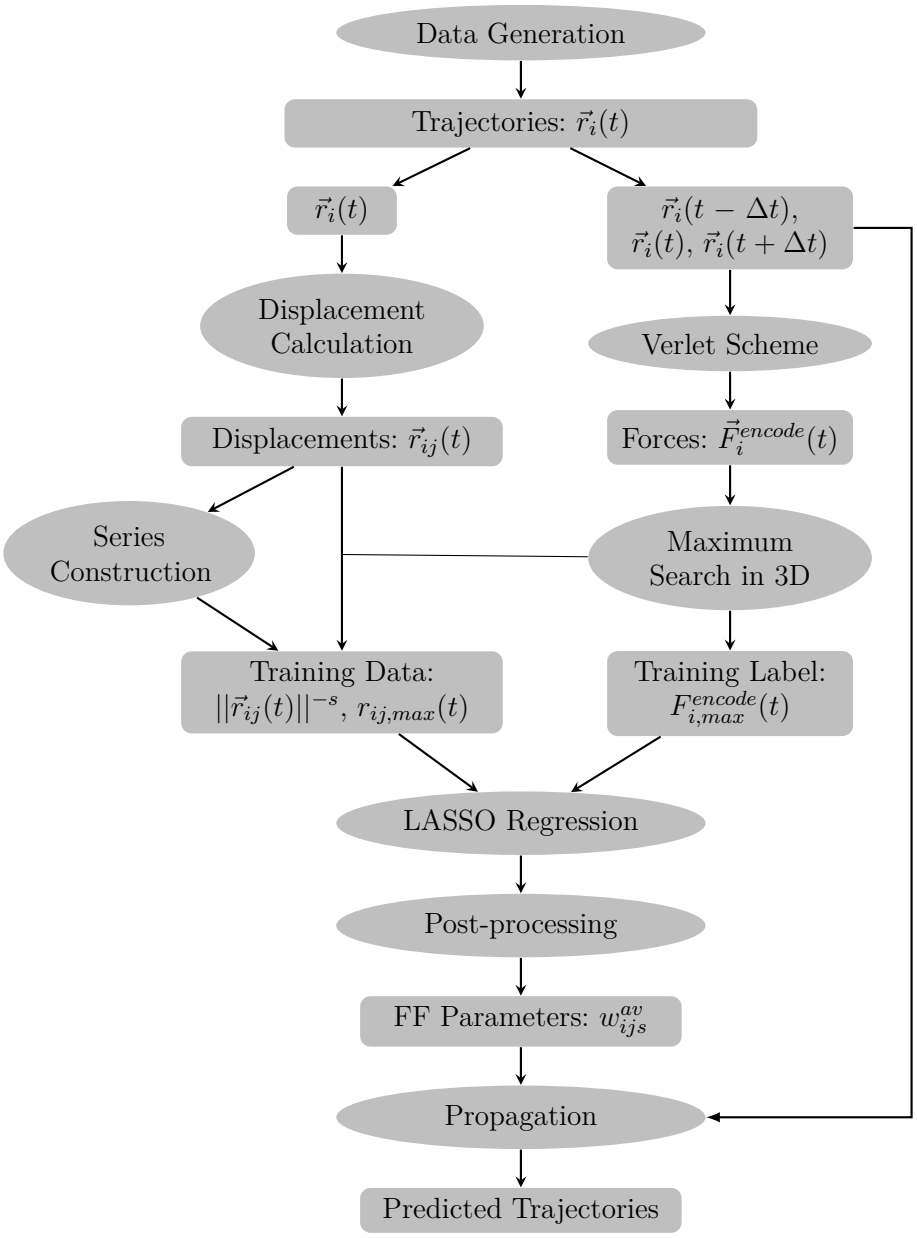


Figure 1. Flowchart of the TrajML MD.

of the atoms i (note that i and later on j denote atom labels rather than atom types) at every time step t from the positions at the three time steps $\vec{r}_i(t-\Delta t)$, $\vec{r}_i(t)$, and $\vec{r}_i(t+\Delta t)$ based on Eq. 1. Then the maximum value $F_{i,max}(t)$ out of the three entries corresponding to the $x/y/z$ directions in $\vec{F}_i(t)$ is selected for each atom i at each step (time t) without an overall preference (see Eq. 7). This step is introduced in the code to avoid calculations of projections on the direction of the movement. $F_{i,max}(t)$ (see Eq. 8) works as the training label at time t ($TL_i(t)$, see Eq. 8) for each atom i. All pairwise displacements of atom i with other atoms are calculated at every time t (denoted as $\vec{r}_{ij}(t)$). Note that the pair i-j here is unique and kept the same throughout the learning and prediction processes. The Euclidean norms of these displacements are used for the construction of the force basis based on a user-defined hyperparameter s. Also, $r_{ij,max}(t)$, i.e. the displacement component in the direction (see Eq. 9) with the maximum value $F_{i,max}(t)$ is selected for each pair i and j (see Eq. 10). $||\vec{r}_{ij}(t)||^{-s}$ and $r_{ij,max}(t)$ form the training data at time t ($TD_{ijs}(t)$) based on Eq. 11. The second term in this equation is a projection of the constructed force basis also on the same direction as the training label at each simulation step (time t). One may notice that only one direction is considered in the constructed $TD_{ijs}(t)$ and $TL_i(t)$ at each time t. Including all three directions in the learning process may lead to inaccurate results when e.g. both $TL_i(t)$ and $TD_{ijs}(t)$ have very small values in any direction, which affects the regression process. This can happen in e.g. linear molecules so we designed the original code in this manner. In other systems, however, we use the information for all three directions as the training data and training label (see Eq. 12 and Eq. 13, we use this form for all examples shown in this manuscript).

$$F_{i,max}^{encode}(t) = \max_{d \in x,y,z} F_{i,d}^{encode}(t) \tag{7}$$

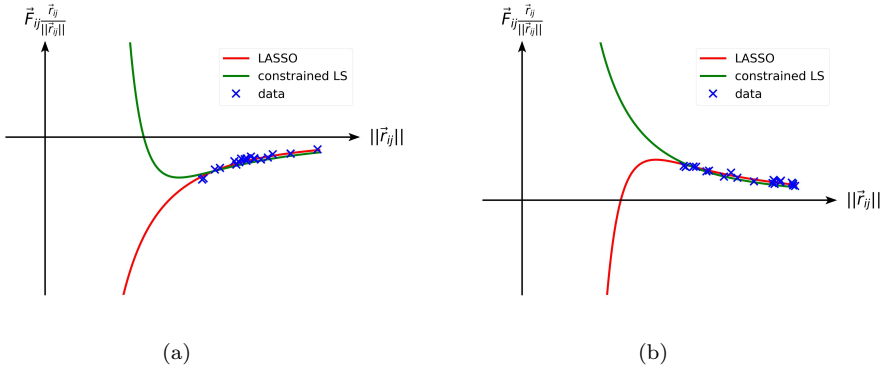
$$TL_i(t) = F_{i,max}^{encode}(t) \tag{8}$$

$$D_i(t) = \underset{d \in x,y,z}{\operatorname{argmax}} F_{i,d}^{encode}(t) \tag{9}$$

$$r_{ij,max}(t) = r_{ij,D_i(t)}(t) \tag{10}$$

$$TD_{ijs}(t) = ||\vec{r}_{ij}(t)||^{-s} \cdot \frac{r_{ij,max}(t)}{||\vec{r}_{ij}(t)||} \tag{11}$$

$$TD_{ijsd}(t) = ||\vec{r}_{ij}(t)||^{-s} \cdot \frac{r_{ij,d}(t)}{||\vec{r}_{ij}(t)||},\ d \in x,y,z \tag{12}$$



(a) (b)

Figure 2. Cases in which nonphysical interactions are produced by LASSO and refitted by constrained LS.

$$TL_{id}(t) = F_{i,d}^{encode}(t),\ d \in x, y, z \tag{13}$$

3.3. Machine learning

LASSO regression is applied on the $TD_{ijsd}(t)$ (x_{ij} in Eq. 6) and $TL_{id}(t)$ (y in Eq. 6) for each atom i regarding all N training samples constructed in the pre-processing with the best model selected by the CV. The force field parameter set w_{ijs} contains the weights obtained from the optimization formula of LASSO shown in Eq. 14. The weight of each atom i is fitted independently, so that the calculation can be efficiently parallelized.

$$w_{ijs} = \underset{\hat{\beta}_{ijs}}{\text{argmin}}\{\frac{1}{N}\sum_{t}[TL_{id}(t) - \frac{1}{m_i}\sum_{j,s} TD_{ijsd}(t)\hat{\beta}_{ijs}]^2 + \lambda\sum_{j,s}|\hat{\beta}_{ijs}|\},\ d \in x, y, z \tag{14}$$

3.4. Post-processing

The post-processing consists of two parts: refitting and averaging.

Refitting is implemented to prevent nonphysical interactions as exemplary shown in Fig. 2. Nonphysical interactions can be obtained from the TrajML due to the lack of short range information between two nonbonding atoms (they are defined as atoms more than three covalent bonds away from each other in this work but can also be defined differently) and thus LASSO regressions overfit the parameters on the long range data (see blue crosses and red lines in Fig. 2). Constrained least squares (LS) regression is adopted for the refitting when the short range repulsion terms are missed in the description of interactions. The refitting only considers two s values in the power series in Eq. 3 (s=2,4 in this work) and the constraint is on the weight of the larger s value (s=4 in this work) so that it is forced to be positive (see green lines in Fig. 2).

Averaging is introduced for the consideration of momentum conservation. In principle, the force field parameter set of atom j to i and atom i to j should be equal (see Eq. 15). However, LASSO regression is carried out for each atom independently, resulting in two parameter sets for interactions between any atom pair i and j.

To fix this, another constrained regression with all atoms included is applied (see Eq. 16). This equation only optimizes nonzero parameters from w_{ijs}^{guess} which represents the

initial guess of the estimated model parameter $\hat{\beta}_{ijs}^{av}$, so that the sparsity of LASSO is kept. The initial guess w_{ijs}^{guess} is obtained by the mass-weighted averaging as given in Eq. 17 on the original parameter set w_{ijs}, where m_i and m_j are the mass of atom i and j, respectively. The equation is proposed based on the idea that the trust level of a parameter set depends on the sampling rate of its training data, which we take to be proportional to the vibrational frequency, and is inversely proportional to the mass of the atom. We also note here that the TrajML only generates a formula of pairwise forces and does not change any time invariant physical laws. Energy conservation is automatically satisfied by TrajML by construction.

$$w_{ijs} \stackrel{!}{=} w_{jis} \tag{15}$$

$$\begin{gathered} w_{ijs}^{av} = \underset{\hat{\beta}_{ijs}^{av} \neq 0}{\operatorname{argmin}} \{ \frac{1}{N} \sum_{t,i} [TL_{id}(t) - \frac{1}{m_i} \sum_{j,s} TD_{ijsd}(t) \hat{\beta}_{ijs}^{av}]^2 \}, \; d \in x, y, z \\ subject\ to\ |\hat{\beta}_{ijs}^{av} - \hat{\beta}_{jis}^{av}| = 0 \end{gathered} \tag{16}$$

$$w_{ijs}^{guess} = \frac{w_{ijs}\sqrt{m_j} + w_{jis}\sqrt{m_i}}{\sqrt{m_j} + \sqrt{m_i}} \tag{17}$$

3.5. Prediction

Both CPU and GPU codes are designed for the prediction of trajectories. Firstly, it reads the atom positions at time t and calculates the pairwise displacements. Then the same series of s as used in the pre-processing is applied on the displacements to generate the force basis $||\vec{r}_{ij}(t)||^{-s}$. The forces at time t are calculated by the element-wise product of the force basis and the parameter sets w_{ijs}^{av} of the constructed force field, followed by a division of m_i (see Eq. 18). Lastly, the Verlet scheme is applied to the atom positions at time step $t - \Delta t$, t, and forces at time step t to predict atom positions at time step $t + \Delta t$ (see Eq. 19, note that we construct the force encoded as $\vec{a}_i(t)\Delta t^2$ as shown in Sec. 2.1).This process is done for each atom i in parallel.

For a system containing N atoms and S power series, as many as $\frac{N(N-1)S}{2}$ parameters are used to describe the force field, and most operations of these parameters are independent from each other. Aiming for the propagation of large systems, GPU as a massive parallel processor is also adopted in this work, and CUDA based packages scikit-cuda [52] and PyCUDA [53] are used.

$$\vec{F}_i^{pred,encode}(t) = \frac{1}{m_i} \sum_{j,s} w_{ijs}^{av} \cdot ||\vec{r}_{ij}(t)||^{-s} \cdot \frac{\vec{r}_{ij}(t)}{||\vec{r}_{ij}(t)||} \tag{18}$$

$$\vec{r}_i(t + \Delta t) = 2\vec{r}_i(t) - \vec{r}_i(t - \Delta t) + \vec{F}_i^{pred,encode}(t) \tag{19}$$

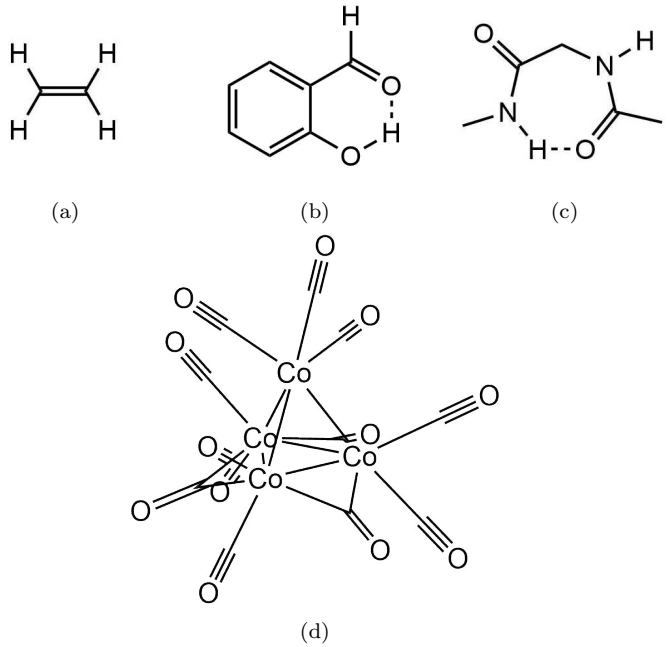


Figure 3. Investigated systems: (a) ethylene (b) salicylaldehyde (c) capped-glycine and (d) tetracobalt dodecacarbonyl.

4. Computational settings

Four systems were investigated in this work: ethylene, salicylaldehyde, capped-glycine, and tetracobalt dodecacarbonyl (see Fig. 3).

DFT-based MD simulations were carried out with the CP2K/QUICKSTEP [54–56] package utilizing Goedecker–Teter–Hutter (GTH) norm-conserving pseudopotentials [57–59], the BLYP [60, 61] exchange–correlation functional, and the corresponding DZVP-GTH basis sets. For equilibration of the system, the NVT ensemble using the Nosé–Hoover thermostat [62, 63] was applied in order to keep the system at the target temperature for 5 ps. Then AIMD in the NVE ensemble was used for the data generation and/or the comparison. The same settings were used for self-consistent-charge density-functional tight-binding [64, 65] (SCC-DFTB)-based MD simulations (denoted as DFTB-based MD simulations in the following). Besides, MD simulations were also performed at the molecular mechanics (MM) level with the OpenMM [66] package and the SMIRNOFF99Frosst force field [67] created by the Open Force Field [68] (OpenFF) Initiative. More detailed information is given in Table 1. A timestep of 0.5 fs was used in all simulations.

AIMD trajectories were used as training data for the TrajML with power series terms $s = 2, 3, 4, 5, 7$ in Eq. 3 and refitting terms $s = 2, 4$ introduced in Sec. 3 for the post-processing. Different sizes of training samples (i.e. No. MD snapshots - 2) of the same system were utilized for comparison (see Table 2). Predictions were then carried out with TrajML FFs learned from trajectories. The number of predicted steps was 50000 for all systems and conditions. For comparison, 50000 steps were also extracted from DFTB- and MM-based MD simulations. Standard deviations of selected vibrational motions were computed to quantify their vibrational behavior in the MD simulations. Mean absolute errors (MAE) of the forces were computed regarding predicted steps using DFT forces as reference. Frequency spectra of different vibrational motions are calculated

Table 1. Overview of simulation settings of the investigated systems. The simulations are labeled according to their purpose training (T) and validation/comparison (C), respectively

System	Simulation level	Temperature [K]	Objective
Ethylene	DFT	300	T & C
Ethylene	DFTB	300	C
Ethylene	MM	300	C
Ethylene	DFT	1000	C
Ethylene ($^{13}C_2D_4$)	DFT	300	C
[Ethylene]$^+$	DFT	300	T & C
[Ethylene]$^+$	DFTB	300	C
Salicylaldehyde	DFT	300	T & C
Salicylaldehyde	DFTB	300	C
Salicylaldehyde	MM	300	C
Capped-glycine	DFT	300	T & C
Capped-glycine	DFTB	300	C
Capped-glycine	MM	300	C
Tetracobalt dodecacarbonyl	DFT	300	T & C

Table 2. Training and predicting settings of the investigated systems.

System	No. training samples	Prediction settings
Ethylene	100, 200, 500, 1000	–
Ethylene	500	velocity rescaled to 1000 K
Ethylene	500	masses changed to ^{13}C and ^{2}H (D)
[Ethylene]$^+$	200, 500, 1000, 2000	–
Salicylaldehyde	500, 1000, 2000, 5000	–
Capped-glycine	5000, 10000, 15000, 20000, 25000	–
Tetracobalt dodecacarbonyl	5000, 10000, 15000	–

individually by the Fourier transform of their values along the MD trajectories. Note that the vibrational motions discussed in this study do not neccessarily directly resemble the vibrational bands that would be obtained from velocity autocorrelation functions of the entire system (power spectrum).

5. Results and Discussions

5.1. Ethylene

The mean values and standard deviations of the C–C' and C–H bond distances, the H–C–H' bond angle, the H–C–C'–H" dihedral angle, and the C'–H'–H–C improper dihedral angle (for labels of the atoms, see Fig. 4) were calculated for different levels of theory in the MD simulations. As Table 3 shows, the DFTB- and MM-MD results were close to those of DFT-MD. We find a good agreement of the structural parameters derived by TrajML (ML100 to ML1000) when compared with the ones obtained by DFT-based MD. Within the given range of testset sizes we observe a steady decrease of the MAE of the force error from 0.250 to 0.116 eV/Å. Vibrational frequencies were also calculated for the above-mentioned five types of bonds/angles using trajectories from the DFT-, DFTB-, MM-, and ML500-based MD simulations (see Fig. 5). The bond stretching vibrations of the C–C' and C–H bonds (obtained by Fourier transform of the respective bond length changes during the AIMD) predicted by TrajML can reproduce the DFT-based MD results with less than 10 % deviation in vibrational frequencies

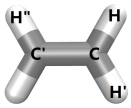


Figure 4. Atomic labels used for ethylene.

Table 3. Comparison of DFT-, DFTB-, MM-, and TrajML-based MD results for the ethylene molecule. MLX represents TrajML using X training samples. Unit: bond length [Å], bond angle/dihedral angle/improper dihedral angle [°], MAE of force [eV/Å].

Type	DFT	DFTB	MM	ML100	ML200	ML500	ML1000
C–C'	1.36±0.02	1.33±0.01	1.35±0.02	1.36±0.03	1.36±0.03	1.36±0.02	1.35±0.03
C–H	1.10±0.03	1.10±0.02	1.08±0.02	1.11±0.04	1.11±0.03	1.10±0.03	1.10±0.03
H–C–H'	116±4	117±4	119±6	117±6	115±5	117±4	117±4
H–C–C'–H"	0±9	0±9	0±9	0±10	0±7	0±6	0±6
C'–H'–H–C	0±6	0±4	0±8	0±4	0±3	0±3	0±3
MAE of force	–	–	–	0.250	0.188	0.120	0.116

and was at the same level of accuracy as the DFTB- and MM-based MD results. The bending motion calculated from the H–C–H' angle changes during the AIMD predicted by TrajML had a frequency shift, but still behaved better than the one obtained with MM. For the H–C–C'–H" dihedral torsional motion and the C'–H'–H–C improper dihedral torsional motion, similar accuracy as those from the DFTB- and MM-based MD results can be achieved.

Besides, we used the TrajML FF of ethylene trained at 300 K for the prediction of its MD trajectory at a temperature of 1000 K. In doing so, slightly larger variations of bond distances and angles and broader bands in the vibrational spectrum at 1000 K compared to the one at 300 K were observed, similar to the observations for DFT-based MD (see Table S1 and Fig. S1).

We also adopted the same TrajML FF trained for ethylene in order to predict the MD trajectory of its isotope molecule $^{13}C_2D_4$. As shown in Fig. S2, the red shift of peaks from ethylene to $^{13}C_2D_4$ seen in the DFT-based MD results was successfully predicted in the ML500 case.

When a positive charge is added to ethylene, it owns a doublet spin state. This kind of system can hardly be described by traditional, standardly available force fields. A reasonable estimation was provided by our TrajML approach (see Table 4) using more than 1000 training samples. There was a transition barrier at 0° of the H–C–C'–H"

Table 4. Comparison of the DFT-, DFTB-, and TrajML-based MD results of [ethylene]$^+$. MLX represents TrajML using X training samples. The absolute value of the H–C–C'–H"* dihedral angle is given. Unit: bond length [Å], bond angle/dihedral angle/improper dihedral angle [°], MAE of force [eV/Å].

Type	DFT	DFTB	ML200	ML500	ML1000	ML2000
C–C'	1.42±0.03	1.35±0.02	1.38±0.02	1.40±0.03	1.41±0.03	1.41±0.03
C–H	1.10±0.03	1.11±0.02	1.11±0.02	1.11±0.03	1.11±0.03	1.11±0.03
H–C–H'	118±5	115±5	115±5	116±5	117±5	117±4
H–C–C'–H"*	28±10	35±12	0±10	-2±8	0±6	7±13
C'–H'–H–C	0±5	0±4	0±3	-1±2	0±3	0±3
MAE of force	–	–	0.343	0.216	0.143	0.143

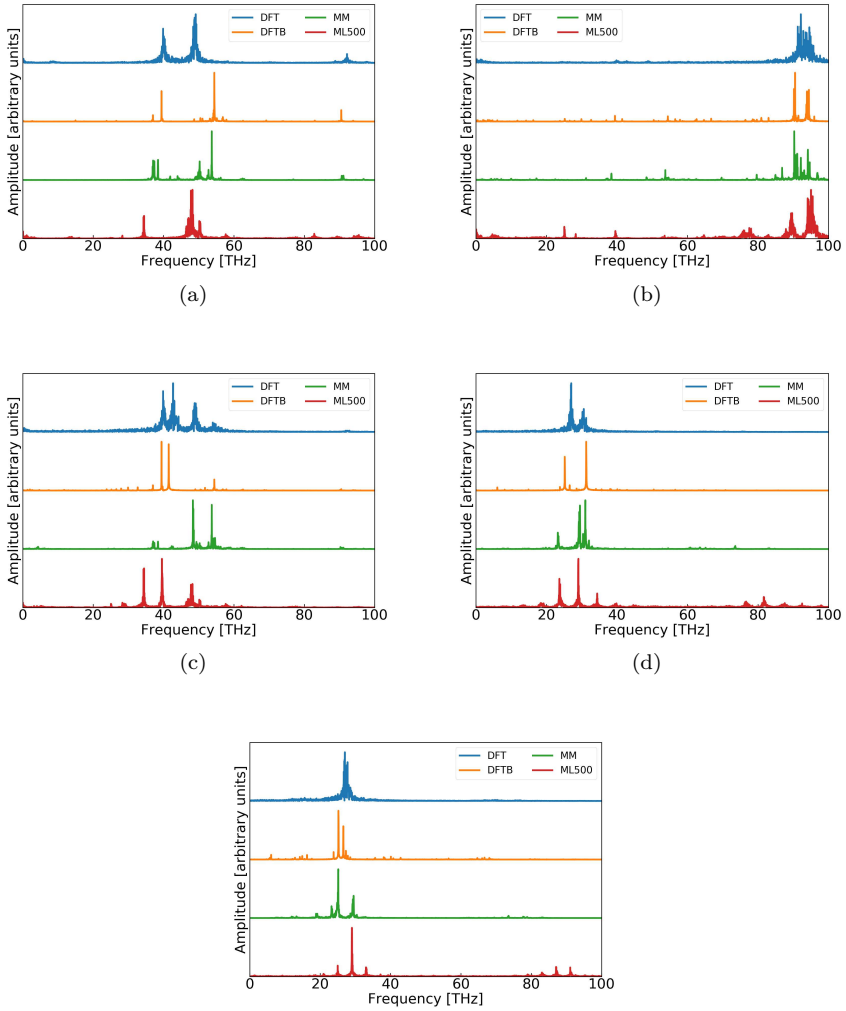


Figure 5. Frequencies obtained from changes of the (a) C−C' bond length, (b) C−H bond length, (c) H−C−H' angle, (d) H−C−C'−H" dihedral angle, and (e) C'−H'−H−C improper dihedral angle of ethylene using the DFT-, DFTB-, MM-, and TrajML-based MD simulations.

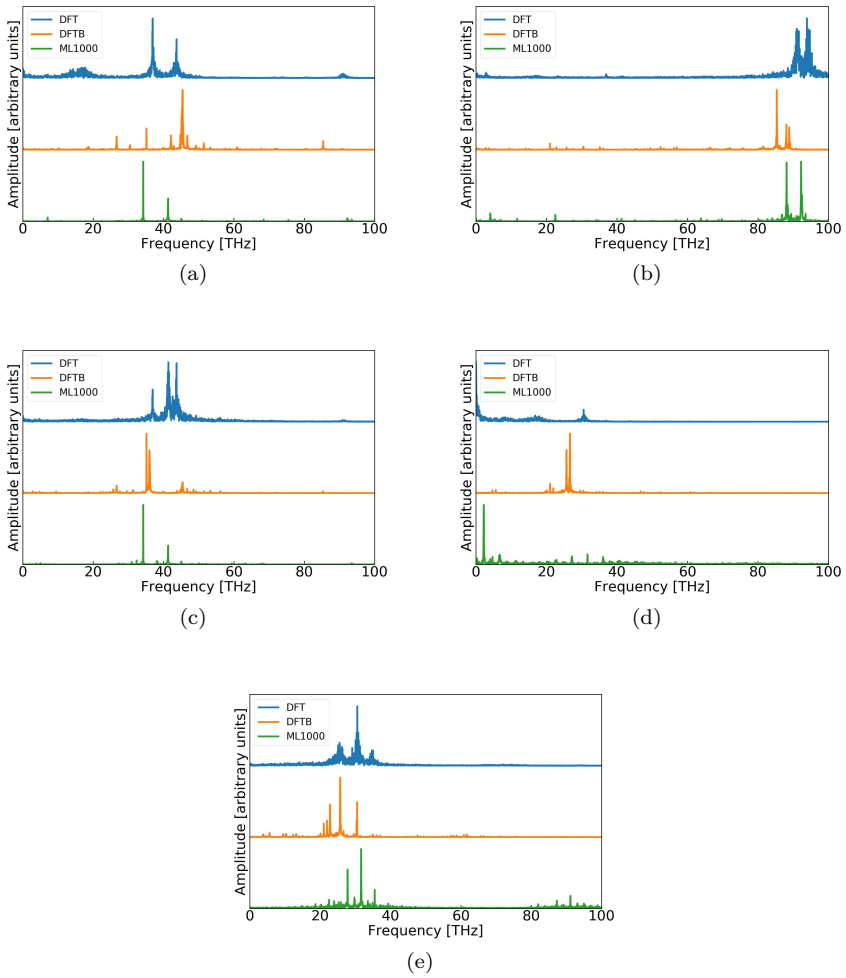


Figure 6. Frequencies obtained from changes of the (a) C−C' bond length, (b) C−H bond length, (c) H−C−H' angle, (d) H−C−C'−H" dihedral angle, and (e) C'−H'−H−C improper dihedral angle of $[\text{ethylene}]^+$ using the DFT-, DFTB-, and TrajML-based MD simulations.

dihedral angle, so it was calculated with its absolute value. Both DFTB and TrajML had a relatively large error for this dihedral angle since the barrier was not well estimated. We note, however, that TrajML can well predict other types of vibrational motions of $[\text{ethylene}]^+$ (see Table 4 and Fig. 6).

5.2. Salicylaldehyde

In order to examine the performance of TrajML for the description of hydrogen bonds, salicylaldehyde was examined (for labels of the atoms, see Fig. 7). Based on DFT-MD for the training, the trajectory of the TrajML-based MD of salicylaldehyde was extracted and compared to the DFT-, DFTB- and MM-based MD simulations. The mean values and the standard deviations of the O' $\cdots$ H hydrogen bond and the C'−C"−O−H and C"−C'−C−O' dihedral angles were calculated for the different simulations. Table 6 shows that the TrajML-based MD simulation gave similar structural results as the DFT-based MD with more than 2000 training samples. Compared to the result from

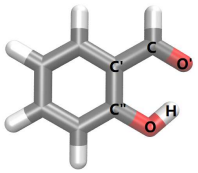


Figure 7. Atomic labels used for salicylaldehyde.

Table 5. Comparison of hydrogen bond length and selected dihedral angles from the DFT-, DFTB-, MM-, and TrajML-based MD simulations of salicylaldehyde. MLX represents TrajML using X training samples. Unit: bond length [Å], dihedral angle [°], MAE of force [eV/Å].

Type	DFT	DFTB	MM	ML500	ML1000	ML2000	ML5000
O' ··· H	1.55±0.12	1.78±0.08	1.86±0.10	1.58±0.14	1.55±0.10	1.56±0.10	1.55±0.14
C'–C"–O–H	0±9	0±10	0±11	0±6	2±6	-1±8	1±5
C"–C'–C–O'	0±10	0±10	0±21	0±8	0±10	0±8	1±9
MAE of force	–	–	–	0.165	0.157	0.153	0.147

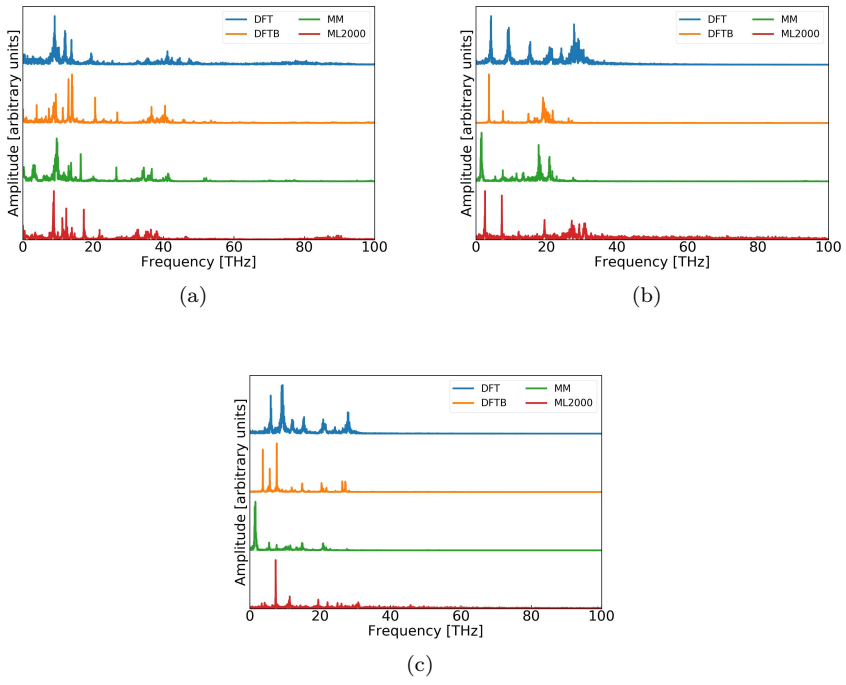


Figure 8. Frequencies obtained from changes of the (a) O' ··· H hydrogen bond distance and the (b) C'–C"–O–H and (c) C"–C'–C–O' dihedral angles of salicylaldehyde using the DFT-, DFTB-, MM-, and TrajML-based MD simulations.

the DFT-based MD, the length of the hydrogen bond O' ··· H was estimated more accurately than with the DFTB-based MD and the variations of the C"–C'–C–O' dihedral angles were estimated more accurately than with the MM-based MD. The comparison in Table 6 and Fig. 8 suggests that the hydrogen bond was reasonably well described by the TrajML FF.

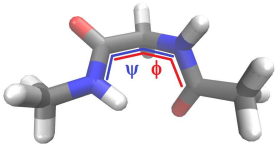


Figure 9. The dihedral angles ψ and ϕ in capped-glycine.

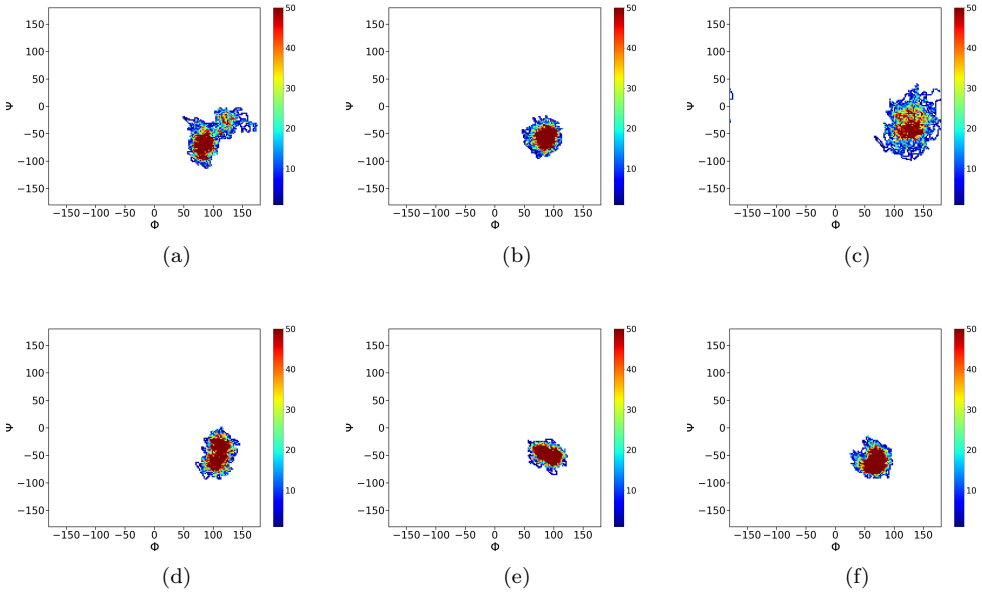


Figure 10. Ramachandran plot of capped-glycine from the (a) DFT-based MD simulation and the TrajML-based MD simulations with (b) 5000, (c) 10000, (d) 15000, (e) 20000, and (f) 25000 training samples.

5.3. Capped-glycine

As another example, capped-glycine was chosen since it features torsional angles which are important for peptides. The DFT-based MD trajectory, which was taken as training data, sampled one of the local minima of the configuration space in which the 7-membered ring was formed (see Fig. 3 (c)). Ramachandran plots of predictions by TrajML are shown in Fig. 10 (b)-(f), from which we can see that at least 20000 training samples were needed to provide a reasonable force field resembling the DFT-MD results (see Fig. 10 (a)), since cases using 10000 and 15000 training samples had biases in the estimation of the local minimum. For further comparison, Ramachandran plots based on the DFTB- and MM-MD trajectories (500000 steps in each case) are provided (see Fig. S3). These two plots show multiple local minima, indicating a limitation of TrajML, namely, potential extrapolation, if the underlying training data is not sufficient. In order to obtain a Ramachandran plot for these local minima by TrajML, trajectories of more initial structures or use of enhanced sampling techniques would be required for the learning of the configurational space by TrajML.

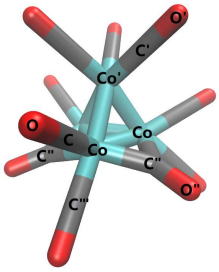


Figure 11. Atomic labels used for tetracobalt dodecacarbonyl.

Table 6. Comparison of DFT- and TrajML-based MD simulations of tetracobalt dodecacarbonyl. MLX represents TrajML using X training samples. Unit: bond length [Å], dihedral angle [°], MAE of force [eV/Å].

Type	DFT	ML5000	ML10000	ML15000
C'–O'	1.16±0.02	1.16±0.02	1.16±0.02	1.16±0.02
C–O	1.16±0.02	1.16±0.02	1.16±0.03	1.16±0.02
C"–O"	1.18±0.02	1.18±0.01	1.18±0.01	1.18±0.01
Co'–C'	1.78±0.04	1.77±0.05	1.77±0.05	1.78±0.05
Co–C	1.76±0.04	1.76±0.04	1.76±0.05	1.76±0.04
Co–C"	1.94±0.04	1.94±0.08	1.95±0.08	1.95±0.09
Co'–Co	2.53±0.08	2.56±0.08	2.55±0.07	2.55±0.07
Co–Co	2.50±0.06	2.49±0.07	2.49±0.07	2.50±0.07
C"–Co–C"	158±4	157±5	157±4	158±5
C–Co–C"'	97±5	98±6	98±5	98±5
Co–C"–Co	80±3	79±3	80±3	80±3
C–Co–Co'–C'	51±9	53±9	51±9	52±8
MAE of force of Co	–	1.190	1.172	1.170
MAE of force of C & O	–	0.134	0.120	0.117

5.4. Tetracobalt dodecacarbonyl

The last system we present here is tetracobalt dodecacarbonyl, a multicenter transition metal complex which can hardly be described by standardly available DFTB or MM. Different types of interactions were compared based on simulation results using DFT- and TrajML-based MD (see Fig. 11 for the atomic labels used; the same label represents the same chemical environment). As shown in Table 6, all types of interactions were reasonably described by the TrajML FF. Three bonding types between C and O and between Co and C can be distinguished based on the estimated mean values of the bond length even though the differences between them were only 0.02 Å in both cases. The variations of the bond length were mostly the same in the TrajML- and the DFT-based MD results except for Co–C", a metal carbonyl bridging bond. It should also be mentioned that the two types of metal–metal bonds (Co'–Co and Co–Co) in the tetracobalt core were also well distinguished with TrajML. The MAE of forces on Co atoms were above 1 eV/Å, which shows a limitation of using only two-body interactions in the, however, description of this system.

5.5. Comparison to other approaches: Ethanol and Uracil

In order to compare the force errors to previously published ones given in Ref. [20], two systems, ethanol and uracil, were also simulated with TrajML. The computational settings were the same as in the cases described above except that the PBE [69] exchange–

Table 7. Force error and speed of prediction using TrajML. Speed is measured on a single CPU core running at 2.2 GHz. Unit: MAE of force [eV/Å], Speed of prediction [frames/s], Training time [min].

System	MAE of force	Speed of prediction	Training time
Ethanol	0.140	7290	0.6
Uracil	0.182	6380	1.0

correlation functional was used. The number of snapshots from the AIMD trajectory were chosen in analogy to the ones used for GDML in Ref. [20]. 1000 geometries were sampled uniformly according to the AIMD trajectory energy distribution and 50000 MD steps were used for the prediction. As shown in Table 7, the MAE of force of ethanol and uracil is 0.140 eV/Å and 0.182 eV/Å using TrajML. For comparison, 0.010 eV/Å and 0.034 eV/Å was achieved using GDML [20] under similar conditions. The error of TrajML can be further reduced by using e.g. a larger training set.

Numbers for the speed of prediction and the training time can be found for sGDML in Ref. [21]. Comparing the speed of prediction for ethanol and uracil, we found 7290 frames/s and 6380 frames/s (see Table 7, using one CPU core of Intel Xeon E5-2650 v4 running at 2.2 GHz) in case of TrajML compared to 826.2 frames/s and 1103.9 frames/s (using Intel Xeon E5-2640 CPU running at 2.4 GHz) for sGDML, respectively. The training time of the same systems using TrajML (0.6 min and 1.0 min) is also less than those using sGDML (2.4 min and 2.0 min). Nevertheless, one should keep in mind the various differences between these approaches. As a comparison, we also performed calculations with the OpenMM [66] MD engine using SMIRNOFF99Frosst force field [67]. Here we obtained a speed of 6930 frames/s and 9860 frames/s (using one CPU core of Intel Xeon E5-2650 v4 running at 2.2 GHz) for uracil and ethanol, respectively.

6. Conclusions

An approach (TrajML) for instant construction of the force fields by trajectory based machine learning is proposed in this work. TrajML is based on the idea that only the nuclear trajectory is required for the training and the prediction of configurations of molecules in MD runs, in this way allowing access to longer MD runs at lower computational effort compared to e.g. the AIMD used for generation of the training data. TrajML can provide a specialized force field designed for the system and the accuracy regarding nuclear structural information in principle can achieve the level of the method it trains from (in our case DFT-MD, but the choice of the method to generate the required MD trajectories is flexible). This alleviates cases where nuclear configurations are of interest for systems that are not parametrized well (e.g. peptides) or can hardly be parametrized (e.g. metal complex in different spin states) by force fields available from databases.

The focus of this manuscript has been on the methodology design of TrajML. In more detail, forces and pairwise displacements have been extracted from the MD trajectory(ies) of the system of interest, and power series of displacements have been used as features for the estimation of forces by LASSO regression, which can alleviate overfitting and ensure the sparsity of the parameter set. Also, constrained LS refitting and averaging have been introduced for the refinement of fitted values. The resulting TrajML FF parameters have then been used to generate more MD trajectories. For the tested examples, TrajML predicted nuclear configurations with an accuracy comparable to

MM- or DFTB-MD. TrajML is also able to drive force fields of reasonable accuracy for systems such as $[ethylene]^+$ and tetracobalt dodecacarbonyl that are inaccessible by MM and DFTB. Moreover, first tests for ethylene regarding the transferability of the TrajML FF have allowed a prediction at different temperature used in the MD or with different atom mass. While the testing examples in this study only use a single plain AIMD trajectory as the training data to demonstrate the general methodology, one would prefer more energy-diversed datasets (e.g. using enhanced sampling for AIMD) as the input for TrajML for pursuing more accurate force fields/potentials. This, however, goes beyond the scope of this work. For more flexible systems, we also suggest multiple simulations with different initial structures and higher kinetic energy as input for the training. This allows a better exploration of the potential energy surface and underlying nuclear structures of the targeted system. Nevertheless, the TrajML method still has short-comings, e.g. 4-body interactions have not been accurately described in some cases, and the capability of potential extrapolation to large configurational space is limited. Also, we note here that the presented approach is targeted on instant system-focused modeling and we are aware that force fields constructed with existing datasets are more general in MD simulations.

Technically speaking, TrajML MD is fast and user-friendly. The MD code using our all-atom force field has achieved more than 2500 frames/s ($\sim$ 106 ns/day) in the prediction for tetracobalt dodecacarbonyl with a single CPU core running at 2.2 GHz. The speed of the TrajML-based MD is in principle similar to that of classical MD packages, since its force field contains only independent distance terms and is thus also suited for parallelization of large systems on GPU. Moreover, force fields are generated automatically and the only input file required is a file with one or more previous trajectories. TrajML MD can be widely adapted to any type of calculations. Unlike traditional force fields and other types of MLFFs/MLPs which have standardly one set of parameters, TrajML method can provide several sets of parameters based on e.g. different QM methods used in the AIMD simulations. The machine learning procedure we have designed directly meets both requirements of energy and momentum conservations without further corrections. More generally, the usage of TrajML MD is not limited to atomic based systems, the trajectory of any system following time-invariant physical laws could be in principle predicted given previous trajectories with user-defined force terms. This is worth to be explored in the long term.

In summary, the proposed TrajML method opens up a new way for the efficient calculation of nuclear configurations in MD simulations and can be based on a variety of data with different accuracy depending on the question of interest. In addition, the use of TrajML is straightforward and user-friendly. This allows access to simulations with longer timescales and for more complex systems than the ones accessible via standardly available, traditional force field-based MD approaches.

7. Supplementary material

See supplementary material for more information regarding the ethylene molecule with different prediction settings (see Table 2) and Ramachandran plots of capped-glycine based on the DFTB- and MM-MD trajectories.

8. Acknowledgments

Funding by the University of Zurich and the Swiss National Science Foundation (grant no: PP00P2_170667) is gratefully acknowledged. We thank the Swiss National Supercomputing Center for computing resources (project ID: s745 and s788).

References

[1] S. Rauscher, V. Gapsys, M.J. Gajda, M. Zweckstetter, B.L. de Groot and H. Grubmüller, J. Chem. Theory Comput. **11** (11), 5513–5524 (2015).
[2] R.S. Paton and J.M. Goodman, J. Chem. Inf. Model. **49** (4), 944–955 (2009).
[3] P. Hohenberg and W. Kohn, Phys. Rev. **136**, B864–B871 (1964).
[4] W. Kohn and L.J. Sham, Phys. Rev. **140**, A1133–A1138 (1965).
[5] J. Behler and M. Parrinello, Phys. Rev. Lett. **98** (14) (2007).
[6] A.P. Bartók, J. Kermode, N. Bernstein and G. Csányi, Phys. Rev. X **8** (4) (2018).
[7] L. Bonati and M. Parrinello, Phys. Rev. Lett. **121** (26) (2018).
[8] A.P. Bartók, M.C. Payne, R. Kondor and G. Csányi, Phys. Rev. Lett. **104** (13) (2010).
[9] V.L. Deringer and G. Csányi, Phys. Rev. B **95** (9) (2017).
[10] G. Hegde and R.C. Bowen, Sci. Rep. **7** (1) (2017).
[11] V. Botu, R. Batra, J. Chapman and R. Ramprasad, J. Phys. Chem. C **121** (1), 511–522 (2016).
[12] Y. Li, H. Li, F.C. Pickard, B. Narayanan, F.G. Sen, M.K.Y. Chan, S.K.R.S. Sankaranarayanan, B.R. Brooks and B. Roux, J. Chem. Theory Comput. **13** (9), 4492–4503 (2017).
[13] R. Jinnouchi, F. Karsai and G. Kresse, Phys. Rev. B **100** (1) (2019).
[14] Z. Li, J.R. Kermode and A.D. Vita, Phys. Rev. Lett. **114** (9) (2015).
[15] K.T. Schütt, F. Arbabzadah, S. Chmiela, K.R. Müller and A. Tkatchenko, Nat. Commun. **8** (1) (2017).
[16] T.L. Fletcher and P.L.A. Popelier, Theor. Chem. Acc. **134** (11) (2015).
[17] T.L. Fletcher and P.L.A. Popelier, J. Comput. Chem. **38** (6), 336–345 (2016).
[18] T.L. Fletcher and P.L.A. Popelier, J. Comput. Chem. **38** (13), 1005–1014 (2017).
[19] J.S. Smith, O. Isayev and A.E. Roitberg, Chem. Sci. **8** (4), 3192–3203 (2017).
[20] S. Chmiela, A. Tkatchenko, H.E. Sauceda, I. Poltavsky, K.T. Schütt and K.R. Müller, Sci. Adv. **3** (5), e1603015 (2017).
[21] S. Chmiela, H.E. Sauceda, I. Poltavsky, K.R. Müller and A. Tkatchenko, Comput. Phys. Commun. **240**, 38–45 (2019).
[22] S. Chmiela, H.E. Sauceda, K.R. Müller and A. Tkatchenko, Nat. Commun. **9** (1) (2018).
[23] K. Yao, J.E. Herr, D.W. Toth, R. Mckintyre and J. Parkhill, Chem. Sci. **9** (8), 2261–2269 (2018).
[24] N. Lubbers, J.S. Smith and K. Barros, J. Chem. Phys. **148** (24), 241715 (2018).
[25] K.T. Schütt, H.E. Sauceda, P.J. Kindermans, A. Tkatchenko and K.R. Müller, J. Chem. Phys. **148** (24), 241722 (2018).
[26] O.T. Unke and M. Meuwly, J. Chem. Theory Comput. **15** (6), 3678–3693 (2019).
[27] F.A. Faber, A.S. Christensen, B. Huang and O.A. von Lilienfeld, J. Chem. Phys. **148** (24), 241717 (2018).
[28] A.S. Christensen, L.A. Bratholm, F.A. Faber and O.A. von Lilienfeld, J. Chem. Phys. **152** (4), 044107 (2020).
[29] H.E. Sauceda, S. Chmiela, I. Poltavsky, K.R. Müller and A. Tkatchenko, J. Chem. Phys. **150** (11), 114102 (2019).
[30] L. Zhang, J. Han, H. Wang, R. Car and W. E, Phys. Rev. Lett. **120** (14) (2018).
[31] A. Glielmo, P. Sollich and A.D. Vita, Phys. Rev. B **95** (21) (2017).
[32] A. Glielmo, C. Zeni and A.D. Vita, Phys. Rev. B **97** (18) (2018).

[33] C. Zeni, K. Rossi, A. Glielmo, Á. Fekete, N. Gaston, F. Baletto and A.D. Vita, J. Chem. Phys. **148** (24), 241739 (2018).
[34] T.D. Huan, R. Batra, J. Chapman, C. Kim, A. Chandrasekaran and R. Ramprasad, J. Phys. Chem. C **123** (34), 20715–20722 (2019).
[35] J. Chapman, R. Batra and R. Ramprasad, Comput. Mater. Sci. **174**, 109483 (2020).
[36] V. Botu, J. Chapman and R. Ramprasad, Comput. Mater. Sci. **129**, 332–335 (2017).
[37] V. Botu, R. Batra, J. Chapman and R. Ramprasad, J. Phys. Chem. C **121** (1), 511–522 (2016).
[38] V. Botu and R. Ramprasad, Int. J. Quant. Chem. **115** (16), 1074–1083 (2014).
[39] T.D. Huan, R. Batra, J. Chapman, S. Krishnan, L. Chen and R. Ramprasad, Npj Comput. Mater. **3** (1) (2017).
[40] A.P. Bartók, S. De, C. Poelking, N. Bernstein, J.R. Kermode, G. Csányi and M. Ceriotti, Sci. Adv. **3** (12), e1701816 (2017).
[41] J. Gilmer, S.S. Schoenholz, P.F. Riley, O. Vinyals and G.E. Dahl, arXiv e-prints arXiv:1704.01212 (2017).
[42] W. Pronobis, A. Tkatchenko and K.R. Müller, J. Chem. Theory Comput. **14** (6), 2991–3003 (2018).
[43] W. Pronobis, K.T. Schütt, A. Tkatchenko and K.R. Müller, Eur. Phys. J. B **91** (8) (2018).
[44] L. Verlet, Phys. Rev. **159** (1), 98–103 (1967).
[45] A.P. Bartók, R. Kondor and G. Csányi, Phys. Rev. B **87** (18) (2013).
[46] A. Glielmo, C. Zeni and A.D. Vita, Phys. Rev. B **97** (18) (2018).
[47] R. Tibshirani, J. Royal Stat. Soc. **58** (1), 267–288 (1996).
[48] M. Stone, J. Royal Stat. Soc. **36** (2), 111–133 (1974).
[49] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss and V. Dubourg, J. Mach. Learn. Res. **12** (Oct), 2825–2830 (2011).
[50] A. Laio and M. Parrinello, Proc. Natl. Acad. Sci. U. S. A. **99** (20), 12562–12566 (2002).
[51] Y. Sugita and Y. Okamoto, Chem. Phys. Lett. **314** (1-2), 141–151 (1999).
[52] L.E. Givon, T. Unterthiner, N.B. Erichson, D.W. Chiang, E. Larson, L. Pfister, S. Dieleman, G.R. Lee, S. van der Walt, B. Menn, T.M. Moldovan, F. Bastien, X. Shi, J. Schlüter, B. Thomas, C. Capdevila, A. Rubinsteyn, M.M. Forbes, J. Frelinger, T. Klein, B. Merry, N. Merill, L. Pastewka, L.Y. Liu, S. Clarkson, M. Rader, S. Taylor, A. Bergeron, N.H. Ukani, F. Wang, W.K. Lee and Y. Zhou, scikit-cuda 0.5.3: a Python interface to GPU-powered libraries 2019, May.
[53] A. Klöckner, N. Pinto, Y. Lee, B. Catanzaro, P. Ivanov and A. Fasih, Parallel Comput. **38** (3), 157–174 (2012).
[54] CP2K version 7.0 (Development Version), the CP2K developers group. CP2K is freely available from https://www.cp2k.org/.
[55] J. Hutter, M. Iannuzzi, F. Schiffmann and J. VandeVondele, Wiley Interdiscip. Rev. Comput. Mol. Sci. **4** (1), 15–25 (2013).
[56] J. VandeVondele, M. Krack, F. Mohamed, M. Parrinello, T. Chassaing and J. Hutter, Comput. Phys. Commun. **167** (2), 103–128 (2005).
[57] S. Goedecker, M. Teter and J. Hutter, Phys. Rev. B **54** (3), 1703–1710 (1996).
[58] C. Hartwigsen, S. Goedecker and J. Hutter, Phys. Rev. B **58** (7), 3641–3662 (1998).
[59] M. Krack, Theor. Chem. Acc. **114** (1-3), 145–152 (2005).
[60] A.D. Becke, Phys. Rev. A **38** (6), 3098–3100 (1988).
[61] C. Lee, W. Yang and R.G. Parr, Phys. Rev. B **37** (2), 785–789 (1988).
[62] S. Nosé, J. Chem. Phys. **81** (1), 511–519 (1984).
[63] W.G. Hoover, Phys. Rev. A **31** (3), 1695–1697 (1985).
[64] M. Elstner, D. Porezag, G. Jungnickel, J. Elsner, M. Haugk, T. Frauenheim, S. Suhai and G. Seifert, Phys. Rev. B **58** (11), 7260–7268 (1998).
[65] D. Porezag, T. Frauenheim, T. Köhler, G. Seifert and R. Kaschner, Phys. Rev. B **51** (19), 12947–12957 (1995).
[66] P. Eastman, J. Swails, J.D. Chodera, R.T. McGibbon, Y. Zhao, K.A. Beauchamp, L.P.

Wang, A.C. Simmonett, M.P. Harrigan, C.D. Stern, R.P. Wiewiora, B.R. Brooks and V.S. Pande, PLOS Comput. Biol. **13** (7), e1005659 (2017).
[67] D.L. Mobley, C.C. Bannan, A. Rizzi, C.I. Bayly, J.D. Chodera, V.T. Lim, N.M. Lim, K.A. Beauchamp, D.R. Slochower, M.R. Shirts, M.K. Gilson and P.K. Eastman, J. Chem. Theory Comput. **14** (11), 6076–6092 (2018).
[68] The Open Force Field Initiative. Open Force Field Initiative. https://openforcefield.org/.
[69] J.P. Perdew, K. Burke and M. Ernzerhof, Phys. Rev. Lett. **77** (18), 3865–3868 (1996).